(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 191 324**
A2

(12) **EUROPEAN PATENT APPLICATION**

(21) Application number: 86100773.0

(22) Date of filing: 21.01.86

(51) Int. Cl.4: G07F 7/10 , H04L 9/02

(30) Priority: 12.02.85 US 700897

(43) Date of publication of application: 20.08.86 Bulletin 86/34

(84) Designated Contracting States: DE FR GB IT

(71) Applicant: International Business Machines Corporation
Old Orchard Road
Armonk, N.Y. 10504(US)

(72) Inventor: Matyas, Stephen Michael
R D 5 Box 19F
Kingston New York 12401(US)

(74) Representative: Grant, Iain Murray
IBM United Kingdom Patent Operations Hursley Park
Winchester, Hants, SO21 2JN(GB)

(54) Offline pin cryptographic validation.

(57) A method of offline personal identification in and to a multiterminal data processing system, the method using an authentication tree with a one-way authentication tree function, a stored global secret key, a stored global verification value of reference, a personal identification number entered directly by the potential user and a personal key and an index position number entered via a card previously issued to the potential user, the index position number representing the tree path for the user to whom the card was issued, by calculating an authentication parameter as a function of the personal key and the personal identification number; mapping the parameter to a verification value using the index position number in the one way function to the root of the tree; comparing the verification value obtained by the mapping with the stored global verification value of reference; and enabling the system in respect of transaction execution if the comparison meets predetermined criteria.

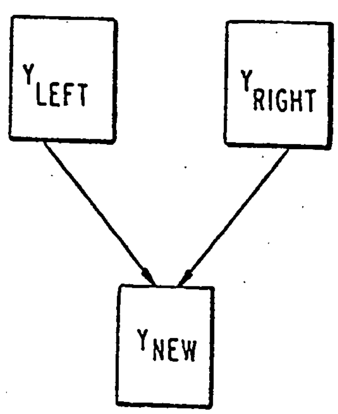


FIG.1

ONE-WAY FUNCTION MAPPING TWO 56-BIT VALUES TO ONE 56-BIT VALUE

EP 0 191 324 A2

OFFLINE PIN CRYPTOGRAPHIC VALIDATION

This invention is directed to a method of offline personal authentication involving a secret user personal identification number (PIN), a secret key and other non-secret data stored on a customer memory card, and a non-secret validation value stored in each terminal connected in a network. Typically, the terminals are connected to a bank which issues the memory card and the terminals are automated teller machines (ATM) or point of sale (POS) terminals.

By "memory card", what is meant is a card which stores more binary data than currently used magnetic strips cards but is distinguished from so-called "smart" cards in that it does not incorporate a microprocessor on the card.

By "offline" is meant, the authentication is not part of the transaction being authenticated. It is usually a separate preliminary operation.

The problem solved by the subject invention is that of authenticating a user of a memory card for electronic funds transfer (EFT) systems or point of sale (POS) terminals. The subject invention is based on a technique of "tree authentication" first suggested by Ralph Merkle. See for example, the following publications:

Ralph C Merkle, "Secrecy, Authentication and Public Key Systems" UMI Research Press, Ann Arbor, Michigan 1982.

Ralph C Merkle, "Secrecy, Authentication and Public Key Systems" Technical Report No 1979-1, Information Systems Laboratory, Stanford University, June 1979.

Ralph C Merkle, "Protocols for Public Key Cryptosystems" Technical Report, BNR, Palo Alto, CA, January 1980.

Ralph C Merkle, "Protocols for Public Key Cryptosystems", Proceedings of the 1980 Symposium on Security and Privacy, 122 - 134 (April 14-16, 1980).

US Patent No 4,300,569 to Ralph C Merkle for "Method of Providing Digital Signatures" discloses a method of providing a digital signature for purposes of authentication of a message. This method utilises an authentication tree function or a one-way function of a secret number. More specifically, the method according to Merkle provides a digital signature of the type which generates a secret number x_i, where x_{i1}, x_{i2}, x_{i3}, . . ., x_{in}, computes $y_i = F(x_i)$ and transmits part of X_i to the receiver as the digital signature. Merkle characterises his invention as providing an authentication tree with an authentication tree function comprising a one-way function of Y_i. The root of the authentication tree and the authentication tree function are authenticated at the receiver. The y_i and the corresponding authentication path values of the authentication tree are transmitted from the transmitter to the receiver. Finally, the y_i are authenticated at the receiver by computing the authentication path of the authentication tree between the y_i and the rest of the authentication tree.

The Merkle method is specifically intended to be an improvement over a public key cryptosystem proposed by Diffie et al in "New Directions in Cryptography", IEEE Transactions on Information Theory, Volume IT-22, Number 6, November 1976, pages 644 to 654, as a means to implement a digital signature and authenticate the true content of a message. In the Diffie et al scheme, to sign a message m whose size is s bits, it is necessary to compute $F(x_1) = Y_1$, $F(x_2) = Y_2$, . . ., $F(x_s) = Y_s$. The transmitter and receiver would agree on the vector $Y = Y_1, Y_2, \ldots, Y_s$. If the jth bit of m was a 1, the transmitter would reveal x_j; but if the jth bit of m was Ø, the transmitter would not reveal x_j. In essence, each bit of m would be individually signed. To avoid the possibility of altering m by the receiver, Diffie et al signed a new message m' that was twice as long as m and computed by concatenating m with the bitwise complement of m. This meant that each bit m_j in the original message was represented by two bits, one of which would not be altered by the receiver.

A major problem of the Diffie et al method addressed by Merkle was that it was only practical between a single pair of users. Accordingly, Merkle's approach provided a signature system of more general application and which rested on the security of a conventional cryptographic function. Moreover, Merkle's authentication tree required less storage than the Diffie et al method. Merkle showed that n values of m bits each could be authenticated on the basis of only $m \times \log_2 (n)$ bits of non-secret information, where "x" denotes multiplication. The one-way function that Merkle envisioned called for a value of m = 1ØØ, although that is not significant in terms of the raw algorithm. The present invention adapts Merkle's idea of tree authentication to the area of offline EFT/POS banking.

It is therefor an object of the present invention to provide an improved offline PIN authentication technique which is particularly adapted for use in EFT/POS terminals.

According to the invention, there is provided a method of offline personal identification in and to a multi-terminal data processing system, the method using an authentication tree with a one-way authentication tree function, a stored global secret key, a stored global verification value of reference, a personal identification number entered directly by the potential user and a personal key and an index position number entered via a card previously issued to the potential user, the index position number representing the tree path for the user to whom the card was issued, by calculating an authentication parameter as a function of the personal key and the personal identification number; mapping the parameter to a verification value using the index position number in the one way function to the root of the tree; comparing the verification value obtained by the mapping with the stored global verification value of reference; and enabling the system in respect of transaction execution if the comparison meets predetermined criteria.

As described, an embodiment comprises a method of offline personal authentication in a multi-terminal system using an improved authenti-cation tree function comprising a one-way function. A person to be authenticated enters his or her PIN and the memory card in a terminal in the multi-terminal system. The information read from the memory card and the PIN are used to calculate an authentication parameter. The calculated authentication parameter is then mapped to a verification value or root of the authentication tree using the one-way function. The verification value obtained by mapping the calculated authentication parameter is then compared with a global verification value stored at the terminal.

In the following description of an embodiment of the present invention, a secure method of tree authentication is realised with a value of m = 56 with the data encryption standard (DES), i.e by making the work factor to break the system equivalent to that of DES key exhaustion. More specifically, if $Y_1, Y_2, \ldots, Y_n$ represents n values to be authenticated by the algorithm, then the global non-secret verification value is calculated via an algorithm that involves all of these n values. With a public key approach, once the public and private key pair has been produced, the secret key can be used to generate the appropriate quantity to store on a memory card without any dependency on the

parameters stored on other memory cards. That is, if an n+1st user is to be added to the list, the public and secret key pair need not be recalculated; rather, they can be used as is to generate the card. But with the DES solution used in the subject invention, an n+1st list user cannot be added to the list without recalculating a new global verification value.

There are ways around this problem. If a bank, for example, is willing to assign a new ID to a customer in cases when his or her PIN and bank card has been compromised, then the original list of n values to be authenticated could include 10 or 20 percent extra IDs and associated values of Y to be authenticated. In that case, when a card and PIN are compromised, the ID is invalidated and a new ID is assigned to the customer and a new PIN and card are issued using one of the precalculated values already available. The old ID is then stored in a "hot list" at each terminal, and in the course of authenticating a user at a terminal, this "hot list" is checked to make sure that the ID being used is not invalid. On the other hand, if the bank cannot assign a new ID to a customer, i.e. the ID remains fixed for the life of that customer, then there can be provided two or more sets of n values and two or more global verification values are stored in the terminal. A user would be assigned a new PIN and a new card to work off the second verification value only if the PIN and card for the first verification value have been compromised. In turn, the user could get a PIN and card to work off a third verification value if the PIN and card for the first two verification values have been compromised. Again, a "hot list" is checked to make sure that the PIN being used and a calculated authentication parameter are not invalid. Yet another possibility is to have only two sets of values, one primary and one secondary. Since there are apt to be very few customers that would be issued more than two cards, these cases could be handled on an exception basis with an authentication table at each EFT/POS terminal. The table, which might contain a few hundred entries, would consist of the user's ID and his authentication parameter, the latter of which would be calculated from the user's PIN, personal key and non-secret data stored on the card, and the global secret key in the terminal.

The method according to the invention also requires a large amount of storage on the card to store non-secret data required by the authentication algorithm. Roughly, if there are 2^n customers that require offline authentication, then each card must store 56 x n bits of non-secret data required by the authentication algorithm. The card must also store a 56 bit secret key and an n-bit number representing the "path" of the calculation. Note that the amount of data stored on the card depends on the number of customers - (i.e. it is dependent on n rather than being independent of it). For example, if there are one million customers in the bank (roughly equal to 2^{20}), then there are 56 x 20 = 1120 bits plus a 56-bit key plus a 20-bit "path" required to be stored on the card. However, the algorithm has the property that the number of bits on the card grows only as the $\log_2$ of the number of customers. Thus, if a bank wants to service two million customers, it is only necessary to store an additional 56 bits on the card. With a memory card having sufficient storage, the DES approach can be used almost as easily as a public key approach. The Rivest, Shamir and Adleman (RSA) public key algorithm, for example, would require a 400-bit value to be stored on the card. This is less than the 1200 or 1300 bits called for by the method according to the present invention, but it is large enough to also require a memory card. In other words, the public key approach can not use the present magnetic strips card either. The approach taken by the present invention has the advantage of offering an alternative to public key and is based on the proven strength of the DES.

The present invention will be described further by way of example with reference to the aforesaid preferred embodiment of the invention as illustrated in the accompanying drawings, in which:-

Figure 1 is a block diagram illustrating the one-way function mapping of two 56-bit values to one 56-bit value;

Figure 2 is a simple illustrative example of three tables of authentication parameters;

Figure 3 is similar to Figure 2 but shows the manner in which values are selected from the three tables to be stored on a customer's memory card; and

Figure 4 is a flow diagram illustrating the operation of the offline PIN validation of this particular embodiment of the invention.

Consider first the question of PIN secrecy. Let the encrypted PIN (denoted EPIN) to be calculated as in Equation 1.

$$EPIN = E_{KGb1}(E_{PIN}(ID)) \quad (1)$$

where PIN is the entered PIN, ID is the user identifier, and KGb1 is a global secret key stored in each EFT/POS terminal. Let the authentication parameter AP be calculated as in Equation 2.

$$AP = Right56[E_{KP \oplus EPIN}(ID) \oplus ID] \quad (2)$$

where KP is the user's personal key stored on the card, the symbol "$\oplus$" represents an Exclusive OR operation, and "Right56" is a function that extracts the rightmost 56 bits in the binary variable denoted by the argument of the function. Equation 2 uses EPIN instead of PIN so that the PIN cannot be derived via trial and error at electronic speeds from a lost or stolen card using the public verification value in the EFT/POS terminal. From Equation 2, it is apparent that a new PIN can be issued merely by calculating a new EPIN using Equation 1, calculating a new KP via the Equation $KP_{new} = EPIN_{new} \oplus KP_{old} \oplus EPIN_{old}$, and reissuing a new card with the new value of KP, i.e. KP_{new}, written on the card.

The method of tree authentication makes use of a binary tree. In a tree with 2^n final elements or "leaves", there are 2^n different "paths" from the root of the tree to each final leaf and therefore n transitions between levels. A tree with n = 3 is shown below:

level 3	X	X	X	X	X	X	X	X
level 2	X		X		X		X	
level 1	X				X			
level Ø	X							

If the left branch is denoted by "Ø" and the right branch by "1", then the tree looks like this:

Index Position	ØØØ	ØØ1	Ø1Ø	Ø11	1ØØ	1Ø1	11Ø	111
level 3	Ø	1	Ø	1	Ø	1	Ø	1
level 2	Ø		1		Ø		1	
level 1	Ø				1			
level Ø	Ø							

The "path" followed in the tree can be represented as a string of "1's" and "Ø's". For example, starting from the root, if we go up to a left branch, then to a right branch, and then to a right branch again, the path is given by the number Ø11. If, on the other hand, we go up to a right branch, and then to a right branch again, and then up to a left branch, the path is given by the number 11Ø. Thus, the numbers ØØØ, ØØ1, . . ., 11 describe the eight paths in this binary tree. It should be apparent that these path numbers also represent the index positions, in binary numbers, of the values at the highest level of the tree. The index position always starts from level zero.

Now, it is assumed that the problem to be resolved is to calculate a single non-secret verification value V from a set of n predefined authentication parameters PA Ø, AP_1, AP_2, . . ., AP_n. Suppose for the sake of this example that $\log_2(n)$ = 2Ø, i.e. that there are $2^{2Ø}$ = 1,Ø58,576 customers. Note that one can always fill in the tree with dummy entries if need be; that is, where the number of customers is not equal to 2^i for some integer i. The n values of AP are mapped to a single root value using a one-way function that involves $\log_2(n)$ iterations. At the first iteration, the n = 1,Ø48,576 values are mapped to 524,388 values, the second iteration maps 524,288 values to 262,144 values, and so on until the 2Øth iteration maps two values to one value. Each application of the one-way function maps two 56-bit values (denoted Y_{left} and Y_{right}) to a single 56-bit value (denoted Y_{new}) as illustrated in Figure 1. A suitable one-way function that maps Y_{left} and Y_{right} to Y_{new} is given by Equation 3.

$$\mathrm{Right56}[Y_{left} \oplus E_{\mathrm{Right56}[Ci]}(Y_{left})] = U$$

$$\mathrm{Right56}[Y_{right} \oplus E_U(Y_{right})] = Y_{new} \qquad (3)$$

where Ci is a 64-bit variable computed using Equation 4 given hereinafter. At the first iteration, Equation 3 is used to map APØ, AP, and a unique codeword C to Y_{new}; i.e. Y_{left} is AP Ø and Y_{right} is AP_1. This output Y_{new} may be denoted $APØ_1$. Then, AP_2, AP_3 and a different codeword C are mapped to $AP_{2,3}$ using Equation 3, and so forth. At the second iteration, $APØ_1$, $AP_{2,3}$ and yet a different codeword C are mapped to $APØ_{1,2,3}$ using Equation 3, and so forth. The operations are fairly simple and straightforward. In all, there are n-1 calculations involving Equation 3. The final 56-bit value so produced is stored in each EFT/POS terminal and is used as a global verification value V.

In the example where n = $2^{2Ø}$, the 1,Ø48,576 values of AP, namely $APØ_1$, $AP_{2,3}$, . . ., $AP_{524286,524287}$, which are produced at the first iteration in that order, are stored in a table at a next level designated Table 19; the 262,144 values $APØ_{1,2,3}$, $AP_{4,5,6,7}$, . . ., $AP_{262140,262141,262142,262143}$ which are produced at the second iteration in that order, are stored in a table at a next level designated Table 18; and so on. Thus, the values in Table 2Ø are processed sequentially using the mapping in Equation 3 to produce the values in Table 19. The values in Table 19 are processed sequentially also using the mapping in Equation 3 to produce the values in Table 18, and so on.

In a simple example where n = 3, only three tables would be required. The values APØAP_1, . . ., AP_7 would be stored in Table 3; the values $APØ_1$, $AP_{2,3}$, $AP_{4,5}$, and $AP_{6,7}$ would be stored in the table at the next level, namely, Table 2; and the values APØ,1,2,3 and $AP_{4,5,6,7}$ would be stored in the table at the next level, namely Table 1, as shown in Figure 2.

Each customer is issued a PIN and a bank card on which is recorded a user identifier ID, a unique secret personal key KP, and other information including information that allows a verification value V to be calculated from that customer's authentication parameter AP. The customer's AP value is a function of PIN, KP, ID, and KGb1 as described above, and is calculated via Equations 1 and 2. In the example given in Figure 2 where n = 3, the other information stored on the bank card necessary to allow a verification value V to be calculated would consist of a 56-bit value selected from each of the three tables, i.e. Table 1, Table 2 and Table 3, and a 3-bit index position of the customer's AP value in Table 3.

The rule for determining which 56-bit values must be selected from Tables 1, 2 and 3 for storage on the bank card depends on the index position of AP in Table 3. If, for example, AP_2 is the authentication parameter to be authenticated, then the 3-bit index position equals Ø1Ø in binary, and the values AP_3, $APØ_{,1}$, $AP_{4,5,6,7}$, and Ø1Ø. represent the necessary information that must be stored on the bank card to allow the verification value V to be calculated. Referring now to Figure 3, there is a diagram illustrating the selected path for obtaining the root or verification value for this tree. The diagram shows the value of the index positions for Tables 1, 2 and 3 and the associated AP value at each such position in each table. Thus, for the given example, the starting index position is Ø1Ø and the value of AP is AP_2. The path traced through the tree is represented by the AP values enclosed in triangles whereas the AP values stored on the bank card are enclosed in rectangles. The rule for selecting the three values AP_3, $AP Ø_{,1}$, and $AP_{4,5,6,7}$ is as follows. Starting with the index position of AP_2, i.e. Ø1Ø, the rightmost bit is inverted and this 3-bit number Ø11 is used as the index position of the AP value selected from Table 3. This results in selecting AP_3, since the index position of AP_3 in Table 3 is just Ø11. For convenience, let the value AP_3 selected from Table 3 be denoted by Y_3 where the subscript on Y is the number of the table. The number Ø11 is now shifted one bit to the right, thus producing Ø1, and the rightmost bit is again inverted, and this 2-bit number ØØ is used as the index position of the AP value selected from Table 2. This results in selecting $APØ_{,1}$, since the index position of $APØ_{,1}$ in Table 2 is just ØØ. For convenience, let the value $APØ_{,1}$ selected from Table 2 be denoted by Y_2. The number ØØ is now shifted one more bit to the right, thus producing Ø, and the rightmost bit is again inverted, and this 1-bit number 1 is used as the index position of the AP value selected from Table 1. This results in selecting $AP_{4,5,6,7}$ since the index position of $AP_{4,5,6,7}$ in Table 1 is just ±1. For convenience let the value $AP_{4,5,6,7}$ selected from Table 1 be denoted by Y_1. Thus, the values Y_3, Y_2, Y_1 and the index position Ø1Ø are the values which would be written on the bank card for the example where the associated AP value is AP_2. In the case where n = 2Ø described above, i.e. where 1,Ø48,576 bank cards are issued to customers, each card would have stored on it the values $Y_{2Ø}$, Y_{19}, . . ., Y_1, and a 2Ø-bit index position in Table 2Ø of the AP value to be authenticated. Thus, the amount of information stored on the bank card is variable and depends on the number of customer AP values to be authenticated and therefore on the size of the authenticated and therefore on the size of the authentication tree so produced.

Referring again to Figure 3, the calculation of the verification value V from AP, Y_3, Y_2, Y_1, and the index position number (Ø1Ø in the example) is as follows. This is the calculation performed in the EFT/POS terminal to authenticate a cardholder. The information on the card is, of course, first read into the EFT/POS terminal. If tbe rightmost bit of the index position is Ø, then Y_{new} is calculated with Equation 3 using as inputs Y_{left} = AP and $Y_{right} = Y_3$. This is the calculation performed in the present example, since the rightmost bit of Ø1Ø is Ø. On the other hand, if the rightmost bit of the index position number is 1, then Y_{new} is calculated with Equation 3 using as inputs $Y_{left} = Y_3$ and Y_{right} = AP; that is, the assignment of values is reversed. Now the index position number is shifted one bit to the right, which in the example illustrated in Figure 3, produces the value Ø1. If the rightmost bit of this shifted number is Ø, then Y_{new} is calculated with Equation 3 using as inputs $Y_{left} = Y_{old}$ and $Y_{right} = Y_1$, where Y_{old} is again the value of Y_{new} produced in the previous step. This is the calculation performed in our present example, since the rightmost bit in the shifted number is Ø. On the other hand, if the rightmost bit of the shifted number is 1, then Y_{new} is calculated with Equation 3 using as inputs $Y_{left} = Y_i$ and $Y_{right} = Y_{old}$. Thus, the index position number stored on the card defines how each value of Y_i, also stored on the card, is to be used in the calculation of Y_{new} using Equation e; i.e. whether it is substituted for Y_{left} or for Y_{right} in Equation 3. Moreover, once this order of substitution has been determined, either rAP or the value of Y_{new} produced at the previous step is substituted for the other parameter Y_{left} or Y_{right}. The value of AP is used only at the first step in the calculation of V whereas a value of Y_{new} is used in all subsequent steps in the calculation of V.

The value C in Equation 3 is derived from the index position number stored on the bank card using the following algorithm. Let Q be a 64-bit constant and KA and KB two constant, non-secret cryptographic keys. Q, KA and KB are stored in each EFT/POS terminal and are universal constants whose values are established by the card issuer. If X_1, X_2, X_3, . . ., $_m$ denotes the index position number on the card, represented in binary, then these m bits are used to calculate the following m values of C: C_1, C_2, . . ., C_m, using Equation 4.

$$C_i = E_{Ki}E_{Ki-1} \ldots E_{K1}(Q) \text{ for } i = 1,2,\ldots,m \quad (4)$$

where

$$K_m = KA \text{ if } X_m = \emptyset$$
$$KB \text{ if } X_m = 1$$

$$K_{m-1} = KA \text{ if } X_{m-1} = \emptyset$$
$$KB \text{ if } X_{m-1} = 1$$

•

•

$$K_1 = KA \text{ if } X_1 = \emptyset$$
$$KB \text{ if } X_1 = 1$$

For example, if the index position number is 1Ø11Ø Ø11Ø1 1ØØØ1 11Ø1Ø, then the following 2Ø values of C are calculated and used with Equation 3 to calculated V:

$$
\begin{aligned}
C1 &= E_{KB}(Q)\\
C2 &= E_{KA}E_{KB}(Q) = E_{KA}(C1)\\
C3 &= E_{KB}E_{KA}E_{KB}(Q) = E_{KB}(C2)\\
&\;\;\vdots\\
C19 &= E_{KB}E_{KA}E_{KB}\cdots E_{KB}E_{KA}E_{KB}(Q) = E_{KB}(C18)\\
C2\emptyset &= E_{KA}E_{KB}E_{KA}E_{KB}\cdots E_{KB}E_{KA}E_{KB}(Q) = E_{KA}(C19)
\end{aligned}
$$

Twenty encryptions are required to calculate the 2Ø values of C for a particular 2Ø-bit index position number. $C_{2\emptyset}$ is used with Equation 3 to make the transition from level 2Ø to level 19 in the tree, C_{19} is used with Equation 3 to make the transition from level 19 to level 18 in the tree, and so forth, there being a different value of C used at each fork in the tree. The reason for using different values of C is because of security. If a constant value of C were used at each fork in the tree, then an adversary could launch a birthday type of attack in which a set of Y_{new} values is calculated by chaining one value after the other until there is a match with one of the actual Y_{new} values in the tree. By opening several accounts, an adversary could collect a fairly large set of such actual values and thus reduce his work factor by using the mentioned attack. However, by forcing different values of C, the attack is thwarted.

For the authentication step at the EFT/POS terminal, assume that the information on the bank card is as follows:

ID	-	User Identifier	
KP	-	Secret Personal Key	56 bits
IPN	-	Non-secret Index Position No	2Ø bits
$Y_{2\emptyset}, Y_{19}, \ldots, Y_1$	-	Non-secret Data to Calculate Y	112Ø bits
VS	-	Verification Selection Number	

The difference between secret and non-secret with regard to card data refers to how that data is treated when it resides somewhere off the card. By definition, the card must be protected if any data stored on the card is defined as secret. Other non-secret data on the card receives the same degree of protection as the secret data. It may be desirable to store a number of verification values and a positive file of PA values in each EFT/POS terminal and to authenticate a card-holder using one of these verification values which is selected on the basis of a verification selection number stored on the cardholder's card or to authenticated the card-holder on the basis of a positive file of AP values. To account for the possibility that some customers will lose their cards or a compromise of either their card or PIN may occur, which will require a new card with a new AP value to be reissued to the card-holder, it

may be desirable to authenticate an AP value associated with a reissued card on the basis of a different verification value V. Each EFT/POS terminal therefore stores a value T, which is interpreted as follows. If the verification selection number VS is less than or equal to T, then the value of Vs is used by the terminal to select the verification value V to be used to authenticate the card-holder's AP value.

Assume that the EFT/POS terminal stores the following:

Q	-	Non-secret Constant	64 bits
KA	-	Non-secret Cryptographic Key	56 bits
KB	-	Non-secret Cryptographic Key	56 bits
V	-	Verification Value	56 bits
KGb1	-	Secret Global Cryptographic Key	56 bits
T	-	Number of Verification Values Stored in Terminal	

It should be noted that there may be multiple verification values depending on the particular implementation.

The steps involved in the authentication process are illustrated in Figure 4. First, the card-holder enters his or her PIN into the EFT/POS terminal. The card-holder also submits his or her bank card to the EFT/POS terminal as depicted in block 1. Then, in block 2, the terminal reads the quantities stored on the card. Before proceeding with any calculations, a "hot list" is checked in block 3 to determine if the ID read from the card is valid. In decision block 4, a determination is made as to whether the Id is valid, and if it is not, then the reject indicator is set in block 5. An ID is invalid if a value equal to the value of the ID is found in the "hot list". Otherwise, the process continues to block 6. At this point, the EPIN is calculated from the Id, PIN and secret KGb1 key using Equation 1. In addition AP is calculated from EPIN, KP and ID using Equation 2. A "hot list", which may be the same "hot list" mentioned above, is also checked to determine if the PA is invalid. The AP is invalid if a value equal to the value of AP is found in the "hot list". If the AP is invalid, then the reject indicator is set in block 5. Otherwise, the process continues to decision block 8 where a determination is made as to whether the verification selection number VS is greater than the value of T stored in the EFT/POS terminal. If it is, then the card--holder is authenticated on the basis of a positive file in block 9 instead of on the basis of a verification value V. Such a file can be implemented by storing in the positive file the values of ID and AP for each such user to be authenticated by the positive file. In decision block 1Ø, a determination is made from the file, and if not, then a reject indicator is set in block 5. More particularly, the card-holder's ID is first used to access and obtain a corresponding AP value stored in the positive file, and the card-holder is then authenticated by comparing this AP of reference value for equality with the AP value calculated om block 6.

Returning to block 8, if the verification selection number is less than or equal to T, then the constants C_1, C_2, . . ., $C_{2Ø}$ are calculated, in that order using Equation 4, from Q, KA, KB, and the index position number (IPN) read from the card, and these generated quantities are stored in a table and later accessed when calculating V. Once the constants C_i have all been calculated, V is calculated from AP, $Y_{2Ø}$, Y_{19}, . . ., Y_1, $C_{2Ø}$, C_{19}, . . ., C_1, and the 2Ø-bit index position number represented by IPN = X_1, X_2, . . ., $X_{2Ø}$ using Equation 3 repeatedly, as follows:

$$\mathrm{Right56}[Y_{left} \oplus E_{\mathrm{Right56}[C2Ø]}(Y_{left})) = U_{2Ø}$$

$$\mathrm{Right56}[Y_{right} \oplus E_{U2\emptyset}(Y_{right})] = Y_{new19}$$

$$\text{where } Y_{left} = AP \text{ and } Y_{right} = Y_{2\emptyset} \text{ if } X_{2\emptyset} = \emptyset$$

$$\text{or } Y_{left} = Y_{2\emptyset} \text{ and } Y_{right} = AP \text{ if } X_{2\emptyset} = 1$$

$$\mathrm{Right56}[Y_{left} \oplus E_{\mathrm{Right56}[C19]}(Y_{left})] = U_{19}$$

$$\mathrm{Right56}[Y_{right} \oplus E_{U19}(Y_{right})] = Y_{new18}$$

$$\text{where } Y_{left} = Y_{new19} \text{ and } Y_{right} = Y_{19} \text{ if } X_{19} = \emptyset$$

$$\text{or } Y_{left} = Y_{19} \text{ and } Y_{right} = Y_{new19} \text{ if } X_{19} = 1$$

$$\mathrm{Right56}[Y_{left} \oplus E_{\mathrm{Right56}[C18]}(Y_{left})] = U_{18}$$

$$\mathrm{Right56}[Y_{right} \oplus E_{U18}(Y_{right})] = Y_{new17}$$

$$\text{where } Y_{left} = Y_{new18} \text{ and } Y_{right} = Y_{18} \text{ if } X_{18} = \emptyset$$

$$\text{or } Y_{left} = Y_{18} \text{ and } Y_{right} = Y_{new18} \text{ if } X_{18} = 1$$

.

.

$$\mathrm{Right56}[Y_{left} \oplus E_{\mathrm{Right56}[C1]}(Y_{left})] = U1$$

$$\mathrm{Right56}[Y_{right} \oplus E_{U1}(Y_{right})] = V$$

$$\text{where } Y_{left} = Y_{new1} \text{ and } Y_{right} = Y_{1} \text{ if } X_{1} = \emptyset$$

$$\text{or } Y_{left} = Y_{1} \text{ and } Y_{right} = Y_{new1} \text{ if } X_{1} = 1$$

The foregoing calculations are made in block 12. The verification selection number is decoded at block 13 to select a particular one of the T global reference values stored at the terminal. Then in decision block 14 a determination is made as to whether the calculated value of V is equal to the particular selected global reference value stored in the terminal. If it is not, then the reject indicator is set in block 5. Otherwise, accept indicator is set in block 11.

Returning briefly to decision block 13, by way of example, let T = 2. Then if the verification selection number is 1, a first global reference value is used in making the determination to authenticate the user. However, if the verification selection number is 2, then a second global reference value is used. As already described with reference to decision block 8, if the verification selection number is greater than 2, the user is authenticated on the basis of a positive file in block 9. Obviously, the numbers chosen here are governed by practical considerations, and those skilled in the art will recognise that these are open to modification.

Summarising, the described method has the following security properties: First, compromising a card does not compromise the PIN. Second, compromising the global secret key does not compromise the PIN nor does it allow someone to forge cards and defraud the system. The process of personal authentication is based on a non-secret global value stored in each EFT/POS terminal. Added PIN protection is achieved through the use of the global secret key also stored in each EFT/POS terminal. Compromising this key does not by itself compromise PINs. The justification for employing a global secret key is that with short PINs, there is no way to maintain PIN secrecy if a user's card is compromised and the EFT/POS terminal stores only non-secret quantities. Although a global secret key has a decided disadvantage, it is better to employ such a key when there is no other alternative to strengthen PIN secrecy, especially when it can be anticipated that many user cards will be lost and thus fall into the hands of potential adversaries. As long as the integrity of the global non--secret verification value in the EFT/POS terminal is maintained, there is no global attach against the system. Even if the integrity of a terminal is compromised, then only that one terminal can be attacked. Since the global secret key does not lead to a global attack against the system, there is less motivation for an opponent to go after it.

As described a "hot list" is required with the procedure according to the invention. THis is no different than what would be required with a public key solution or with a DES solution involving only a global secret key for user authentication. The "hot list" is needed because the bank has to have a way to invalidate an account. For example, an opponent could open an account under a phony name and then proceed to duplicate his card and sell the cards and PINs for profit.

A user's PIN can be changed, but this involves reissuing the customer's bank card. Basically, when the PIN is changed, compensating changes must be made on the bank card which involves recalculation of an offset or certain non-secret parameters on the card. If a user's card and PIN have been compromised, then a new card and PIN must be issued. In this case, an entry on the "hot list" must be made to effectively invalidate the authentication information stored on that card and the user's PIN. This does not necessarily mean that the ID is invalidated. The method is such that a customer's assigned ID can remain the same even if a new card and PIN are issued, although it is more efficient if a new ID is issued.

While the invention has been described in terms of a preferred embodiment in the environment of a banking multi-terminal network, those skilled in the art will recognise that the principles of the invention can be practiced in other environments where it is desired to provide for the offline personal authentication of users of a system. For example, the invention could be used in a security system that would allow access to secure areas only to users of the system who are properly authenticated at a terminal. The important feature of the invention is the use of an authentication tree with an authentication tree function comprising a one-way function.

Claims

1. A method of offline personal identification in and to a multiterminal data processing system, the method using an authentication tree with a one-way authentication tree function, a stored global secret key, a stored global verification value of reference, a personal identification number entered directly by the potential user and a personal key and an index position number entered via a card previously issued to the potential user, the index position number representing the tree path for the user to whom the card was issued, characterised in that

a)an authentication parameter is calculated as a function of the personal key and the personal identification number;

b)mapping the parameter to a verification value using the index position number in the one way function to the root of the tree;

c)comparing the verification value obtained by the mapping with the stored global verification value of reference; and

d)enabling the system in respect of transaction execution if the comparison meets predetermined criteria.

2.A method as claimed in claim 1 wherein the step of mapping is performed by first calculating a different code word for each node of the authentication tree and then using the different code words at the iteration of each node.

3. A method as claimed in claim 2 further comprising the step of storing, in each terminal, values of Q, an m-bit constant, and KA and KB, two non-secret cryptographic keys, the calculation of a different codeword for each node being a function of Q, KA and KB and the index position number stored on the user's card.

4. A method as claimed in claim 3 wherein the step of calculating an authentication.parameter is performed by the steps of:-

calculating an encrypted personal identification number (PIN), denoted EPIN, by the equation

$$EPIN = E_{KGb1}(E_{PIN}(ID))$$

where KGb1 is a global secret key stored in each terminal and ID is a user identification; and

calculating an authentification parameter AP by the equation

$$AP = Right56[E_{KP \oplus EPIN}(ID) \oplus ID]$$

where KP is the user's personal key stored on the card, the symbol "⊕" represents an Exclusive OR operation, and "Right56" is a function that extracts the rightmost 56 bits in the binary variable denoted by the argument of the function.

5. A method as claimed in claim 4 wherein there is further

stored, on the card, m values $Y_1, Y_2, \ldots, Y_m$, to be authenticated and the step of mapping is performed by the step of calculating the verification value V from AP, the m values and the tree function by the equations

$$\mathrm{Right56}[Y_{left} \oplus E_{\mathrm{Right56}[Ci]}(Y_{left})] = U$$

$$\mathrm{Right56}[Y_{right} \oplus E_U(Y_{right})] = Y_{new}$$

where Y_{left} and Y_{right} are two values in the tree path and C_i are different values of the code word calculated for each iteration at each node of the tree function and the last Y_{new} in the iteration is the verification value V.

6. A method as claimed in claim 5 wherein the values of Ci are calculated by the equation

$C_i = E_{Ki}E_{Ki-1} \ldots E_{K1}(Q)$ for $i = 1,2,\ldots,m$ where K_i = KA if $X_i = \emptyset$ and K_i = KB if $X_i = 1$ and $X_1, X_2, X_3, \ldots, X_m$ denote binary bits stored on the user's card which represent the index position number.

7. A method as claimed in any of claims 4 to 6 wherein the user identifier ID is additionally stored on the user's card, the method further comprising the step of checking the user identifier read from the user's card against a list to determine if the ID is invalid, and, if it is, inhibiting the transaction.

8. A method as claimed in claim 7 wherein, after calculating the authentification parameter, a list is checked to determine if the authentification parameter is invalid and, if it is so, inhibiting the transaction.

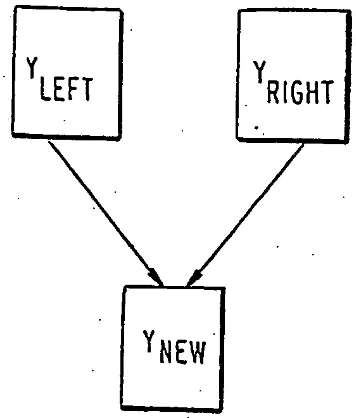


FIG.1

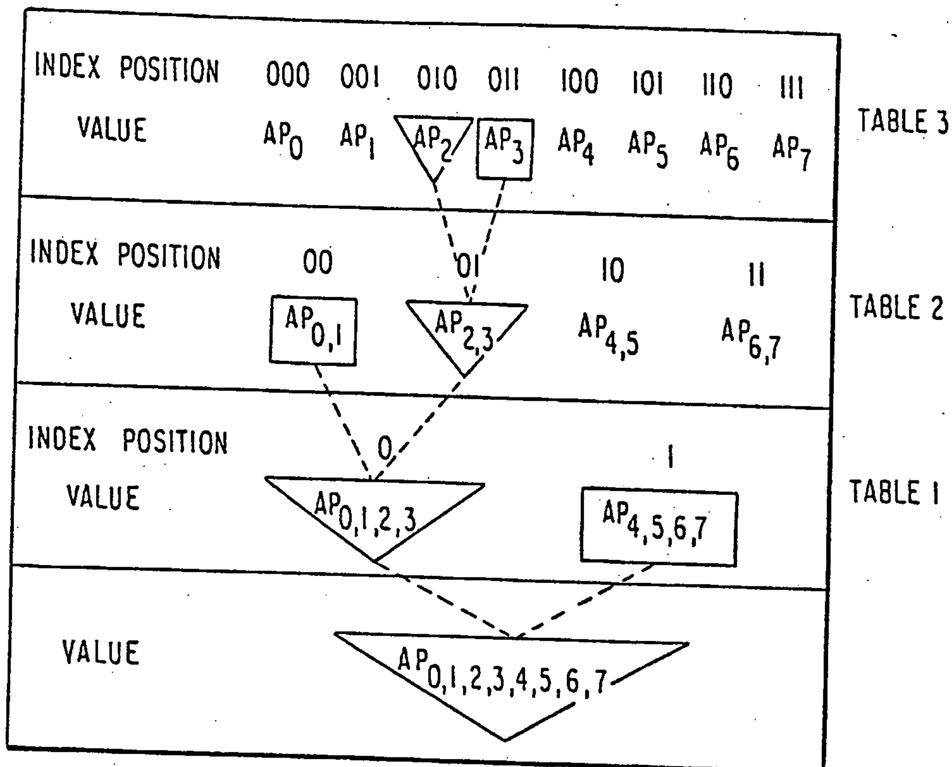
ONE-WAY FUNCTION MAPPING TWO 56-BIT VALUES TO ONE 56-BIT VALUE

FIG.2

INDEX POSITION	000	001	010	011	100	101	110	111	TABLE 3
VALUE	AP_0	AP_1	AP_2	AP_3	AP_4	AP_5	AP_6	AP_7	
INDEX POSITION	00		01		10		11		TABLE 2
VALUE	$AP_{0,1}$		$AP_{2,3}$		$AP_{4,5}$		$AP_{6,7}$		
INDEX POSITION	0				1				TABLE 1
VALUE	$AP_{0,1,2,3}$				$AP_{4,5,6,7}$				
VALUE	$AP_{0,1,2,3,4,5,6,7}$								

NOTE: INDEX POSITION IS REPRESENTED AS A BINARY NUMBER.

VALUES STORED IN TABLES 1,2 AND 3 WHERE n = 3.



NOTE : INDEX POSITION IS REPRESENTED AS A BINARY NUMBER

FROM TABLE	INDEX POSITION OF VALUE SELECTED
3	011
2	00
1	1

VALUES SELECTED FROM TABLES 1, 2 AND 3 TO BE STORED ON CUSTOMER'S MEMORY CARD FOR THE PARTICULAR CASE WHERE THE CUSTOMER'S AUTHENTICATION PARAMETER IS AP_2

FIG. 3

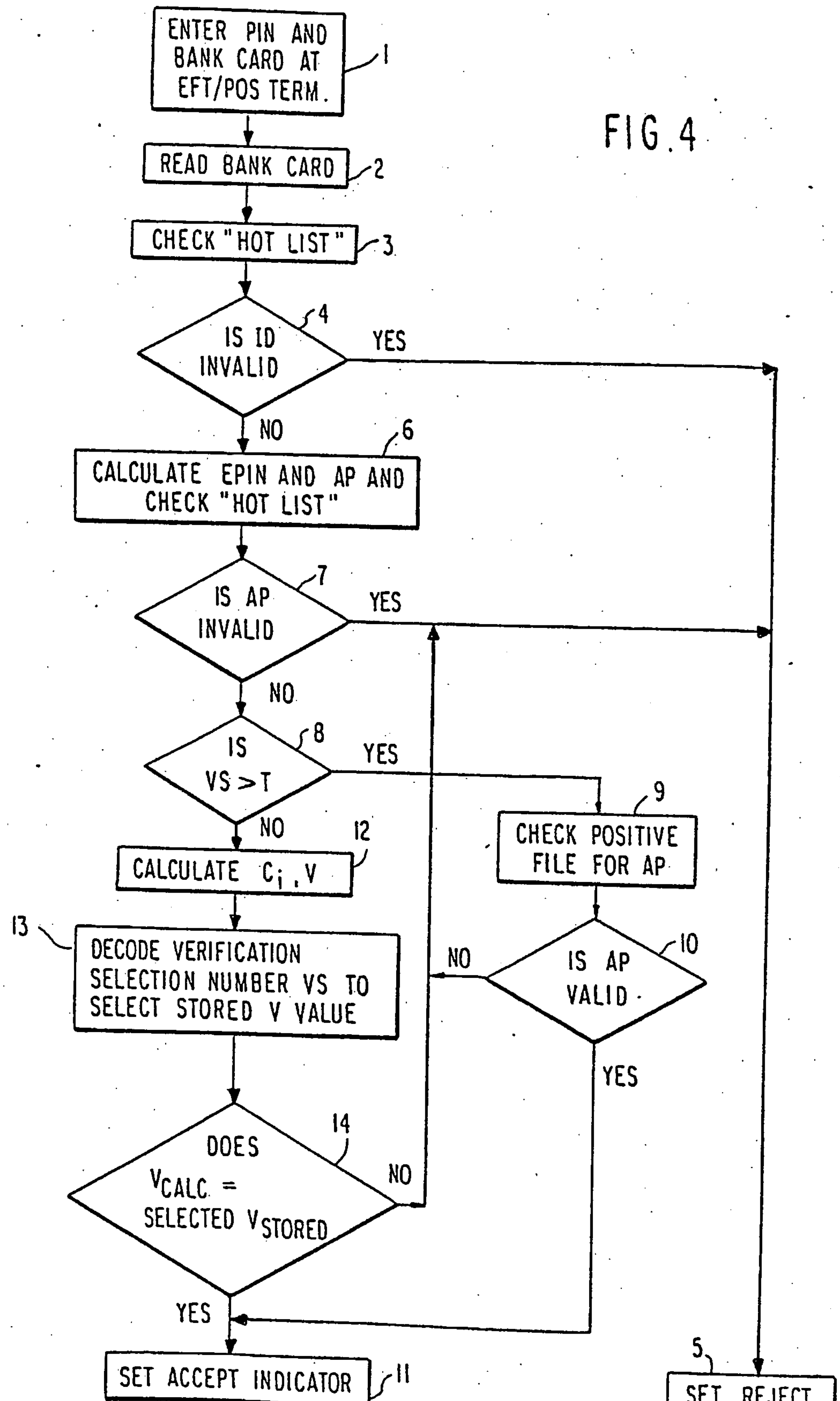

ENTER PIN AND BANK CARD AT EFT/POS TERM.
1
READ BANK CARD
2
CHECK "HOT LIST"
3
IS ID INVALID
4
YES
NO
CALCULATE EPIN AND AP AND CHECK "HOT LIST"
6
IS AP INVALID
7
YES
NO
IS VS > T
8
YES
NO
CHECK POSITIVE FILE FOR AP
9
IS AP VALID
10
NO
YES
CALCULATE Ci, V
12
DECODE VERIFICATION SELECTION NUMBER VS TO SELECT STORED V VALUE
13
DOES VCALC = SELECTED VSTORED
14
NO
YES
SET ACCEPT INDICATOR
11
SET REJECT INDICATOR
5

FIG.4